Arduino

Racing game

Mateusz Tobor

Spis treści

Strona

Temat projektu

Gra wyścigowa z czarno-białą grafiką 2D.

HOLE RACING	**Nazwa gry:** HOLE RACING **Fabuła:** Jadąc autostradą omiń wszystkie dziury w drodze! Wjechanie w dziurę powoduje ogromne uszkodzenia samochodu, a co za tym idzie - GAME OVER!

O projekcie

W projekcie zastosowano czarno-biały wyświetlacz do którego zaimplementowano funkcję automatycznego włączania podświetlenia po zmroku, oraz wyłączania takiego podświetlenia, gdy jest jasno.

Niesamowitą wygodę użytkowania zapewnia sterowanie pilotem IR, a dodatkowe wrażenia zapewnia dźwięk, który dostarczany jest przez buzzer.

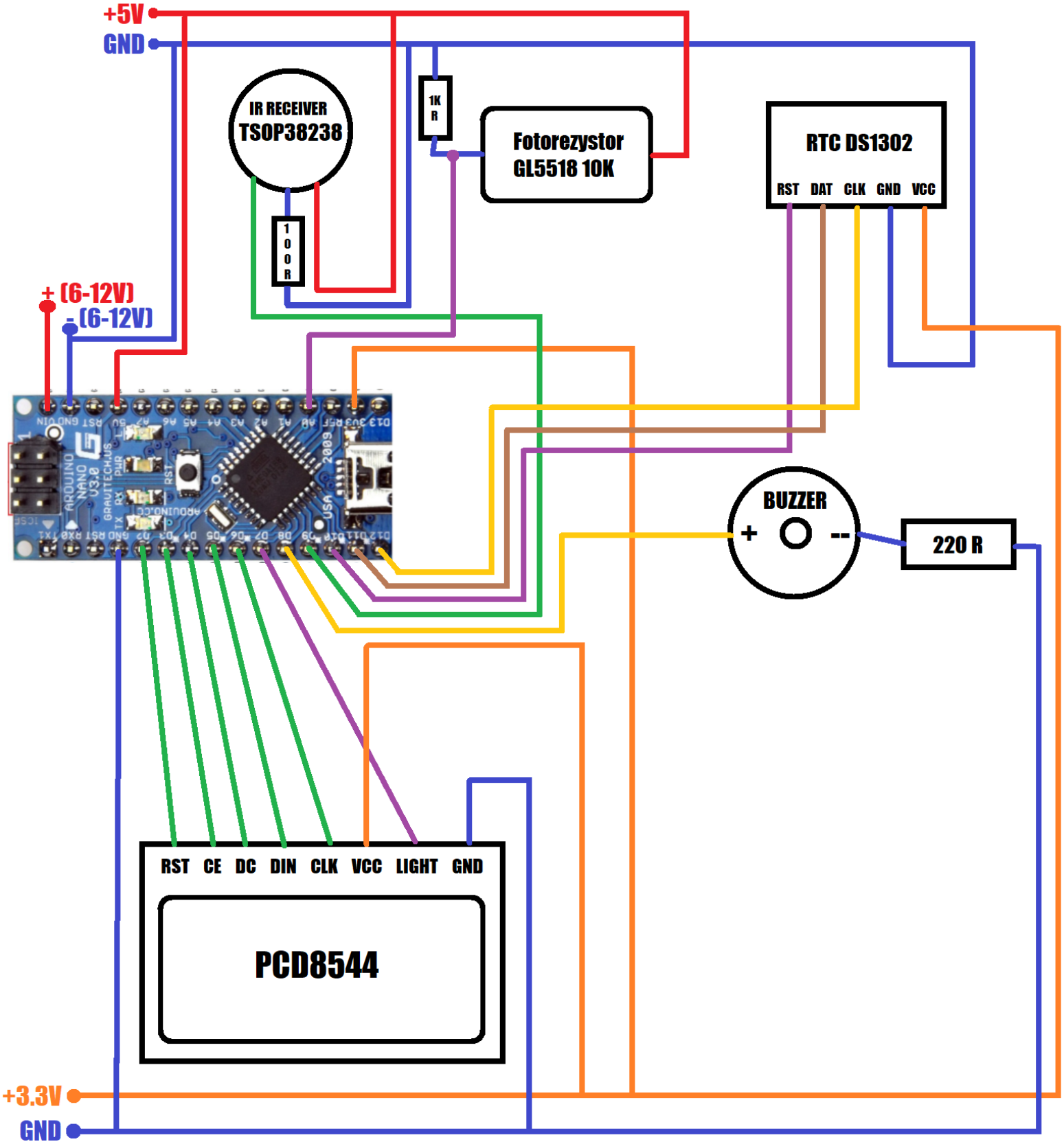
Specyfikacja techniczna

- Wyświetlacz: czarno-biały
- Rozdzielczość wyświetlacza: 84x48
- Sterowanie: przy pomocy pilota IR
- Dźwięk: tak
- Dodatkowe funkcje:
    - czujnik światła - automatyczne podświetlenie wyświetlacza

Zastosowane komponenty

- Arduino NANO v3
- Wyświetlacz zgodny z PCD8544
- Odbiornik podczerwieni TSOP38238
- Moduł RTC DS1302
- Pasywny buzzer
- Fotorezystor GL5518 10K
- Rezystor 1K R
- Rezystor 220 R
- Rezystor 100 R

Schemat połączeniowy

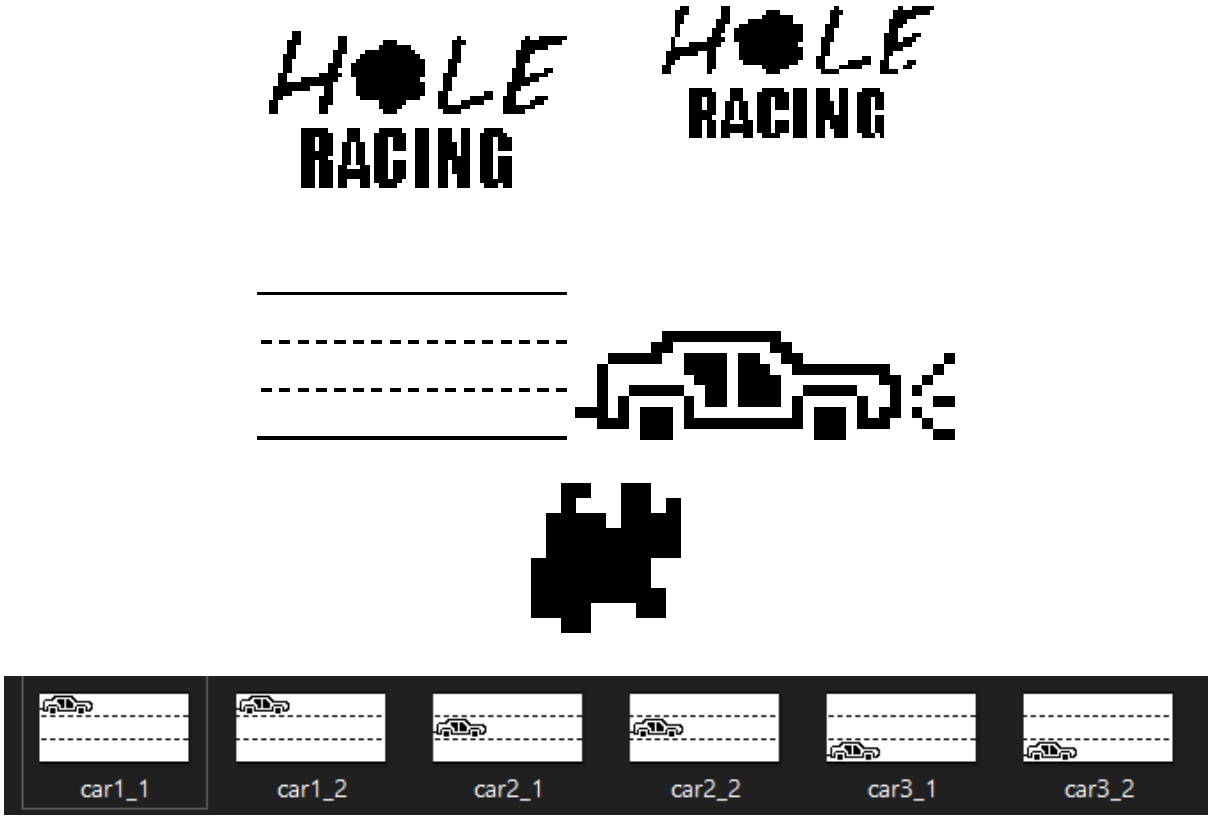


Wykorzystane biblioteki

- **NOKIA5110_TEXT**
  https://github.com/gavinlyonsrepo/NOKIA5110_TEXT
- **ezBuzzer**
  https://www.arduino.cc/reference/en/libraries/ezbuzzer/
- **IR Remote**
  https://github.com/Arduino-IRremote/Arduino-IRremote
- **virtuabotixRTC**
  https://github.com/chrisfryer78/ArduinoRTClibrary/blob/master/virtuabotixRTC.h
- **Arduino**
- **TimeLib**

Zastosowana grafika (monochromatic bitmap)

Grafika zastosowana w tym projekcie została stworzona specjalnie na potrzeby projektu.



Kod programu Arduino

main

```
//DEFINE PINS
#define PIN_LCD_RST         2     //LCD RST PIN
#define PIN_LCD_CE          3     //LCD CE PIN
#define PIN_LCD_DC          4     //LCD DC PIN
#define PIN_LCD_DIN         5     //LCD DIN PIN
#define PIN_LCD_CLK         6     //LCD CLOCK PIN
#define PIN_LCD_LIG         7     //LCD LIGHT PIN
#define PIN_LIGHT_SENSOR    0     //LIGHT SENSOR PIN
#define PIN_BUZZER          8     //BUZZER PIN
#define PIN_IR_RECEIVER     9     //IR RECEIVER PIN
#define PIN_RTC_CLK         12
#define PIN_RTC_DAT         11
#define PIN_RTC_RST         10

//CONFIG
#define CONFIG_LCD_inverse  false
#define CONFIG_LCD_contrast 0xBF // default is 0xBF set in LCDinit, Try
0xB1 - 0xBF if your display is too dark/dim
#define CONFIG_LCD_bias 0x13 // LCD bias mode 1:48: Try 0x12 , 0x13 or
0x14

//IR CONTROL CODES
#define IRC_RESET 16711935
#define IRC_MUTE 16744575
#define IRC_MANUAL_LIGHT_ENABLE 16728255
#define IRC_MANUAL_LIGHT_SET 16736415
#define IRC_START 16746615
#define IRC_TOP 16756815
#define IRC_BOT 16754775

//LOAD LIBS
#include <Arduino.h>
#include "TimeLib.h"
#include <NOKIA5110_TEXT.h> //LCD
#include <ezBuzzer.h>       //SOUND
#include <IRremote.h>       //IR
#include <virtuabotixRTC.h> //REAL TIME

//CREATING OBJECTS
NOKIA5110_TEXT lcd(PIN_LCD_RST, PIN_LCD_CE, PIN_LCD_DC, PIN_LCD_DIN,
PIN_LCD_CLK);
ezBuzzer buzzer(PIN_BUZZER);
IRrecv ir(PIN_IR_RECEIVER);
decode_results irr;
virtuabotixRTC rt(PIN_RTC_CLK, PIN_RTC_DAT, PIN_RTC_RST);

void setup() {
  Serial.begin(9600);
```

```
  //seconds, minutes, hours, day of the week, day of the month, month,
year
  //rt.setDS1302Time(00, 59, 18, 1, 4, 4, 2022);

  //DISPLAY
  pinMode(PIN_LCD_LIG, OUTPUT);
  DISPLAY_SET_LIGHT(0);
  lcd.LCDInit(CONFIG_LCD_inverse, CONFIG_LCD_contrast, CONFIG_LCD_bias);
// init  the LCD
  lcd.LCDClear(0x00);
  lcd.LCDFont(1);

  //IR
  ir.enableIRIn();
  //ir.blink13(false);

  //BUZZER
  pinMode(PIN_BUZZER, OUTPUT);//buzzer


  //welcome screen
  DISPLAY_SET_LIGHT(1);
  bmp_load();
  lcd.LCDClear(0x00);//hole Racing
  DISPLAY_SET_LIGHT(0);
}
bool mute = false;
bool display_light = false;
bool display_manual_light_enable = false;
bool init_menu = false;
bool started_game = false;
bool init_start_game = false; //always false on start
bool game_over = false;
int car_position = 2;
int holes[3][2] = {{0,0},  //level 1
                   {0,0},  //level 2
                   {0,0}}; //level 3
bool anim = true;
char score[5];
int score_tmp;

//time for score
tmElements_t time_start;
time_t time_start_unix=0;
tmElements_t time_now;
time_t time_now_unix=0;
tmElements_t time_lh;
time_t time_lh_unix=0;

int holes_speed=0;
int max_holes=0;
```

```
void loop() {
  buzzer.loop();
  IR_CONTROL();
  DISPLAY_LIGHT();


  if(started_game) {
    if(game_over) _GAME_OVER();
    else {
      //INIT START GAME
      if(!init_start_game) {
        buzzer.stop();
        car_position = 2;
        _score_set_timestart();//set timer start
        lcd.LCDFont(6);
        init_start_game = true;
      }
      lcd.LCDClear(0x00);
      draw_car();
      if(!mute) sound_car();
      anim = !anim;
      _score();
      generate_holes();
      draw_holes();
      delay(70);
    }
  } else {
    if(!init_menu) {
      bmp_menu();
      init_menu=!init_menu;
    }
    if(!mute) sound_menu();
  }

}
```

bitmap

```
const uint8_t PROGMEM logo_Bitmap[] = {
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x80, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0xf0, 0xfe, 0x1f, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0,
0xf8, 0x7f, 0x0f, 0x00, 0xc0, 0xe0, 0xf0, 0xf8, 0xf8, 0xf8, 0xfc, 0xfe,
0xfe, 0xfe, 0xfc, 0xfc, 0xfc, 0xf8, 0xf0, 0xe0, 0x00, 0x00, 0x00, 0x00,
0x00, 0xe0, 0xfe, 0x3f, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x80, 0xf8, 0xfe, 0x3f, 0x0e, 0x06, 0x07, 0x87, 0x83,
0x03, 0x03, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xf8,
0x3f, 0x1f, 0x1c, 0x0c, 0x0c, 0x0e, 0x0e, 0x06, 0x06, 0xe7, 0xff, 0x7f,
0x03, 0x01, 0x00, 0x00, 0x01, 0x1f, 0x3f, 0x3f, 0x3f, 0x3f, 0xff, 0xff,
0xff, 0xff, 0xff, 0x7f, 0x3f, 0x3f, 0x3f, 0x1d, 0x00, 0x00, 0x00, 0xf0,
0xff, 0x8f, 0x81, 0x80, 0xc0, 0xc0, 0x60, 0x60, 0x60, 0x30, 0x30, 0x00,
0x00, 0xe0, 0xfc, 0xff, 0x8f, 0x87, 0x86, 0xc3, 0x63, 0x61, 0x21, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x01,
0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xf0, 0xf0, 0x30, 0xf1, 0xf1, 0xe0,
0x00, 0x00, 0x80, 0xf0, 0x70, 0xf0, 0x80, 0x00, 0x00, 0xe0, 0xf0, 0xf1,
0x31, 0xf1, 0xf0, 0xe0, 0x00, 0x00, 0xf0, 0xf0, 0xf0, 0x00, 0x00, 0xf0,
0xf1, 0xf1, 0x81, 0x01, 0xf0, 0xf0, 0x00, 0x00, 0xe0, 0xf0, 0xf0, 0x30,
0xf0, 0xe1, 0xc1, 0x01, 0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0xff, 0xff, 0x06, 0xff, 0xff, 0xfb,
0x00, 0xf0, 0xff, 0x6f, 0x60, 0x6f, 0xff, 0xf0, 0x00, 0xff, 0xff, 0xff,
0x80, 0xf9, 0xf9, 0x79, 0x00, 0x00, 0xff, 0xff, 0xff, 0x00, 0x00, 0xff,
0xff, 0x07, 0x3f, 0xfc, 0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0xff, 0x80,
0xfd, 0xfd, 0xfd, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x01, 0x01, 0x00, 0x01, 0x01, 0x01,
0x00, 0x01, 0x01, 0x00, 0x00, 0x00, 0x01, 0x01, 0x00, 0x00, 0x01, 0x01,
0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x01, 0x01, 0x01, 0x00, 0x00, 0x01,
0x01, 0x00, 0x00, 0x01, 0x01, 0x01, 0x00, 0x00, 0x00, 0x01, 0x01, 0x01,
0x00, 0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

const uint8_t PROGMEM menu_Bitmap[] = {
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0xfc, 0x1e, 0x06, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0,
0xf0, 0x7e, 0x1e, 0x00, 0xc0, 0xe0, 0xe0, 0xf0, 0xf0, 0xf0, 0xf8, 0xfc,
0xfc, 0xfc, 0xf8, 0xf8, 0xf8, 0xe0, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,
```

```
0xe0, 0xfc, 0x3e, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x80, 0xf0, 0xfc, 0x3e, 0x1c, 0x0c, 0x8e, 0x86, 0x06, 0x06, 0x02,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0xc0, 0xf8, 0x3f, 0x1c, 0x0c, 0x0c, 0x0e, 0x0e, 0x06, 0x06, 0xff, 0x7f,
0x03, 0x01, 0x00, 0x00, 0x01, 0x1f, 0x3f, 0x3f, 0x3f, 0x3f, 0x7f, 0xff,
0xff, 0xff, 0x7f, 0x7f, 0x3f, 0x3f, 0x1d, 0x00, 0x00, 0x00, 0x70, 0xff,
0x8f, 0x81, 0x80, 0x40, 0x40, 0x60, 0x60, 0x60, 0x30, 0x30, 0x00, 0x00,
0xe0, 0xff, 0x8f, 0x87, 0x06, 0x43, 0x63, 0x21, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xf8, 0xf8, 0x18, 0xf8, 0xf0,
0x00, 0x00, 0xc0, 0xf8, 0x38, 0xf8, 0xc0, 0x00, 0x00, 0xf0, 0xf8, 0xf8,
0x18, 0xf8, 0xf8, 0xf0, 0x00, 0xf8, 0xf8, 0xf8, 0x00, 0x00, 0xf8, 0xf8,
0xf8, 0xc0, 0x00, 0xf8, 0xf8, 0x00, 0x00, 0xf0, 0xf8, 0xf8, 0x18, 0xf8,
0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0x3f, 0x3f, 0x03, 0x3f, 0x3d,
0x00, 0x3c, 0x3f, 0x1b, 0x18, 0x1b, 0x3f, 0x3c, 0x00, 0x1f, 0x3f, 0x3f,
0x20, 0x3c, 0x1c, 0x1c, 0x00, 0x3f, 0x3f, 0x3f, 0x00, 0x00, 0x3f, 0x3f,
0x03, 0x0f, 0x3e, 0x3f, 0x3f, 0x00, 0x00, 0x1f, 0x3f, 0x3f, 0x20, 0x1e,
0x3e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

const uint8_t PROGMEM car1_1_Bitmap[] = {
  0x01, 0x01, 0xe1, 0x11, 0x11, 0x91, 0x91, 0x99, 0x8d, 0x85, 0x35, 0x75,
0xf5, 0xf5, 0x05, 0xf5, 0xf5, 0xe5, 0xcd, 0x19, 0x31, 0xa1, 0xa1, 0xa1,
0xa1, 0xa1, 0x21, 0x21, 0x61, 0xc1, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
  0x02, 0x22, 0x27, 0x24, 0x07, 0x00, 0x2e, 0x2e, 0x2e, 0x00, 0x07, 0x24,
0x25, 0x25, 0x04, 0x05, 0x25, 0x25, 0x25, 0x04, 0x07, 0x20, 0x2e, 0x2e,
0x0e, 0x00, 0x27, 0x24, 0x26, 0x03, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00,
```

```
0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20,
0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00,
0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20,
0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00,
0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04,
0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00,
0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
  0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80
};

const uint8_t PROGMEM car1_2_Bitmap[] = {
  0x01, 0x01, 0xe1, 0x11, 0x11, 0x91, 0x91, 0x99, 0x8d, 0x85, 0x35, 0x75,
0xf5, 0xf5, 0x05, 0xf5, 0xf5, 0xe5, 0xcd, 0x19, 0x31, 0xa1, 0xa1, 0xa1,
0xa1, 0xa1, 0x21, 0x21, 0x61, 0xc1, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
  0x22, 0x22, 0x07, 0x04, 0x27, 0x20, 0x2e, 0x0e, 0x0e, 0x20, 0x27, 0x24,
0x05, 0x05, 0x24, 0x25, 0x25, 0x05, 0x05, 0x24, 0x27, 0x20, 0x0e, 0x0e,
0x2e, 0x20, 0x27, 0x04, 0x06, 0x23, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20,
0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00,
0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20,
0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00,
```

```
0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04,
0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00,
0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00,
  0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80
};

const uint8_t PROGMEM car2_1_Bitmap[] = {
  0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
  0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0xa0, 0x80, 0x80, 0xa0,
0xa0, 0xa0, 0x80, 0x80, 0xa0, 0xa0, 0xa0, 0x00, 0x00, 0x20, 0x20, 0x20,
0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00,
0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20,
0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00,
0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20,
0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
  0x40, 0x40, 0xfc, 0x82, 0xe2, 0x12, 0xd2, 0xd3, 0xd1, 0x10, 0xe6, 0x8e,
0xbe, 0xbe, 0x80, 0xbe, 0xbe, 0xbc, 0xb9, 0x83, 0xe6, 0x14, 0xd4, 0xd4,
0xd4, 0x14, 0xe4, 0x84, 0xcc, 0x78, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x05, 0x05, 0x05, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x05, 0x05,
0x01, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00,
0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04,
0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00,
0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
  0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80
};
```

```
const uint8_t PROGMEM car2_2_Bitmap[] = {
  0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
  0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x80, 0xa0, 0xa0, 0xa0,
0x80, 0x80, 0xa0, 0xa0, 0xa0, 0x80, 0x80, 0x20, 0x20, 0x20, 0x00, 0x00,
0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20,
0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00,
0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20,
0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00,
  0x40, 0x40, 0xfc, 0x82, 0xe2, 0x12, 0xd2, 0xd3, 0xd1, 0x10, 0xe6, 0x8e,
0xbe, 0xbe, 0x80, 0xbe, 0xbe, 0xbc, 0xb9, 0x83, 0xe6, 0x14, 0xd4, 0xd4,
0xd4, 0x14, 0xe4, 0x84, 0xcc, 0x78, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x05, 0x01, 0x01, 0x04, 0x04, 0x04,
0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x01, 0x01,
0x05, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04,
0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00,
0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00,
  0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80
};

const uint8_t PROGMEM car3_1_Bitmap[] = {
  0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
  0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20,
0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00,
0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20,
```

```
0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00,
0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20,
0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x04, 0x84, 0x44, 0x40, 0x40, 0x44, 0x64, 0x34, 0x10, 0xd0, 0xd4,
0xd4, 0xd4, 0x10, 0xd0, 0xd4, 0x94, 0x34, 0x60, 0xc0, 0x84, 0x84, 0x84,
0x80, 0x80, 0x84, 0x84, 0x84, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00,
0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04,
0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00,
0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
  0x88, 0x88, 0x9f, 0x90, 0x9c, 0x82, 0xba, 0xba, 0xba, 0x82, 0x9c, 0x91,
0x97, 0x97, 0x90, 0x97, 0x97, 0x97, 0x97, 0x90, 0x9c, 0x82, 0xba, 0xba,
0xba, 0x82, 0x9c, 0x90, 0x99, 0x8f, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80
};

const uint8_t PROGMEM car3_2_Bitmap[] = {
  0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01,
  0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00,
0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20,
0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00,
0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20,
0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20,
0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00, 0x20, 0x20, 0x20, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x04, 0x04, 0x80, 0x40, 0x44, 0x44, 0x44, 0x60, 0x30, 0x14, 0xd4, 0xd4,
0xd0, 0xd0, 0x14, 0xd4, 0xd4, 0x90, 0x30, 0x64, 0xc4, 0x84, 0x80, 0x80,
0x84, 0x84, 0x84, 0x80, 0x80, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04,
```

```
0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00,
0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04,
0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04,
0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00, 0x04, 0x04, 0x04, 0x00, 0x00,
  0x88, 0x88, 0x9f, 0x90, 0x9c, 0x82, 0xba, 0xba, 0xba, 0x82, 0x9c, 0x91,
0x97, 0x97, 0x90, 0x97, 0x97, 0x97, 0x97, 0x90, 0x9c, 0x82, 0xba, 0xba,
0xba, 0x82, 0x9c, 0x90, 0x99, 0x8f, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80
};

const uint8_t PROGMEM hole_Bitmap[] = {
  0xe0, 0xfc, 0xff, 0xfd, 0xfc, 0xf8, 0xff, 0xff, 0x9c, 0x1e,
  0x01, 0x01, 0x03, 0x03, 0x00, 0x00, 0x00, 0x01, 0x01, 0x00
};

void bmp_load() {
  lcd.LCDClear(0x00);
  lcd.LCDgotoXY(0, 0);
  lcd.LCDCustomChar(logo_Bitmap, sizeof(logo_Bitmap) / sizeof(unsigned
char), 0x00, true);
  delay(2500);
  lcd.LCDClear(0x00);
  lcd.LCDgotoXY(0, 0);
  lcd.LCDString("Autorzy: ");
  lcd.LCDFont(6);
  lcd.LCDgotoXY(0, 2);
  lcd.LCDString("Mateusz Tobor");
  lcd.LCDgotoXY(0, 3);
  lcd.LCDString("Kaja Nowicka");
  lcd.LCDgotoXY(0, 4);
  lcd.LCDString("Kuba Rychlicki");
  delay(3000);
  lcd.LCDClear(0x00);
  lcd.LCDFont(1);
  lcd.LCDgotoXY(4, 1);
  lcd.LCDString("UNIWERSYTET");
  lcd.LCDgotoXY(23, 2);
  lcd.LCDString("SLASKI");
  lcd.LCDgotoXY(0, 4);
  lcd.LCDString("w Katowicach");
  delay(2000);
  lcd.LCDClear(0x00);
}

void bmp_menu() {
  lcd.LCDgotoXY(0, 0);
  lcd.LCDCustomChar(menu_Bitmap, sizeof(menu_Bitmap) / sizeof(unsigned
char), 0x00, true);
```

```
  lcd.LCDFont(6);
  lcd.LCDgotoXY(8, 4);
  lcd.LCDString("WCISNIJ START");
  lcd.LCDgotoXY(0, 5);
  lcd.LCDString("ABY ROZPOCZAC GRE");
}

void draw_car() {
  switch(car_position) {
    lcd.LCDgotoXY(0, 1);
    case 1:
      if(anim) lcd.LCDCustomChar(car1_1_Bitmap, sizeof(car1_1_Bitmap) /
sizeof(unsigned char), 0x00, true);
      else lcd.LCDCustomChar(car1_2_Bitmap, sizeof(car1_2_Bitmap) /
sizeof(unsigned char), 0x00, true);
    break;

    case 2:
      if(anim) lcd.LCDCustomChar(car2_1_Bitmap, sizeof(car2_1_Bitmap) /
sizeof(unsigned char), 0x00, true);
      else lcd.LCDCustomChar(car2_2_Bitmap, sizeof(car2_2_Bitmap) /
sizeof(unsigned char), 0x00, true);
    break;

    case 3:
      if(anim) lcd.LCDCustomChar(car3_1_Bitmap, sizeof(car3_1_Bitmap) /
sizeof(unsigned char), 0x00, true);
      else lcd.LCDCustomChar(car3_2_Bitmap, sizeof(car3_2_Bitmap) /
sizeof(unsigned char), 0x00, true);
    break;

    default:
    break;
  }
}

void bmp_hole(int x, int y) {
  lcd.LCDgotoXY(x,y);
  lcd.LCDCustomChar(hole_Bitmap, sizeof(hole_Bitmap) / sizeof(unsigned
char), 0x00, true);
}

void draw_holes() {
  if(holes[0][0] != 0) {
    if(holes[0][1] == 5) bmp_hole(65,24);
    if(holes[0][1] == 4) bmp_hole(61,24);
    if(holes[0][1] == 3) bmp_hole(51,24);
    if(holes[0][1] == 2) bmp_hole(41,24);
    if(holes[0][1] == 1) bmp_hole(31,24);
  }
  if(holes[1][0] != 0) {
```

```
    if(holes[1][1] == 5) bmp_hole(65,26);
    if(holes[1][1] == 4) bmp_hole(61,26);
    if(holes[1][1] == 3) bmp_hole(51,26);
    if(holes[1][1] == 2) bmp_hole(41,26);
    if(holes[1][1] == 1) bmp_hole(31,26);
  }
  if(holes[2][0] != 0) {
    if(holes[2][1] == 5) bmp_hole(65,28);
    if(holes[2][1] == 4) bmp_hole(61,28);
    if(holes[2][1] == 3) bmp_hole(51,28);
    if(holes[2][1] == 2) bmp_hole(41,28);
    if(holes[2][1] == 1) bmp_hole(31,28);
  }
}
```

functions

```
void(* resetFunc) (void) = 0;

void _GAME_OVER() {
    buzzer.stop();

    holes[0][0] = 0;
    holes[0][1] = 0;
    holes[1][0] = 0;
    holes[1][1] = 0;
    holes[2][0] = 0;
    holes[2][1] = 0;

    lcd.LCDClear(0x00);
    if(!mute) sound_lost();
    lcd.LCDInit(!CONFIG_LCD_inverse, CONFIG_LCD_contrast,
CONFIG_LCD_bias);
    lcd.LCDFont(1);
    lcd.LCDgotoXY(10, 2);

    lcd.LCDString("GAME OVER");

    if(!mute) {
      for(int i=0; i<50; i+=1) {
        buzzer.loop();
        delay(100);
      }
    }
    else //delay(4000);

    lcd.LCDClear(0x00);
    lcd.LCDInit(CONFIG_LCD_inverse, CONFIG_LCD_contrast,
CONFIG_LCD_bias);

    game_over=false;
    started_game=false;
    init_menu=false;
    init_start_game=false;
}


void MUTE() {
  mute = !mute;
  buzzer.stop();
}

void generate_holes() {
  if(score_tmp > 20) {
    holes_speed = 1;
    _holes_timeupdate();
    max_holes = 2;
```

```
}
if(score_tmp > 20) {
  holes_speed = 3;

}
else if(score_tmp > 10) {
  holes_speed = 2;

}
else if(score_tmp > 1) {
  holes_speed = 1;
  _holes_timeupdate();
}
else if(score_tmp > 0) {
  max_holes = 1;
}

if(score_tmp < 3) {
  max_holes = 1;
  holes_speed = 0;
  _holes_timeupdate();
}
else if(score_tmp < 12) {
  holes_speed = 1;
  _holes_timeupdate();
}
else if(score_tmp < 20) {
  holes_speed = 2;
  _holes_timeupdate();
}
else if(score_tmp < 30) {
  holes_speed = 2;
  max_holes = 1;
  _holes_timeupdate();
}

if(_holes_gettime() < holes_speed) {

    if(holes[0][0] != 0) {
      if(holes[0][1] > 1) holes[0][1]--;
      else if(car_position == 1) game_over=true;
      else holes[0][0] = 0;
    }
    if(holes[1][0] != 0) {
      if(holes[1][1] > 1) holes[1][1]--;
      else if(car_position == 2) game_over=true;
      else holes[1][0] = 0;
    }
    if(holes[2][0] != 0) {
      if(holes[2][1] > 1) holes[2][1]--;
```

```
        else if(car_position == 3) game_over=true;
        else holes[2][0] = 0;
      }

      _holes_timeupdate();
  }
  if(_holes_count() < max_holes)
      _generate_holes();
}

void _generate_holes() {
  int r = random(0, 3);
  holes[r][0] = 1;
  holes[r][1] = 5;
}

void _holes_timeupdate() {
  rt.updateTime();
  time_lh.Second = rt.seconds;
  time_lh.Hour = rt.hours;
  time_lh.Minute = rt.minutes;
  time_lh.Day = rt.dayofmonth;
  time_lh.Month = rt.month;
  time_lh.Year = rt.year;
  time_lh_unix =  makeTime(time_lh);
}

int _holes_gettime() {
  return (time_now_unix - time_lh_unix);
}

int _holes_count() {
  int c=0;
  if(holes[0][0] != 0) c++;
  if(holes[1][0] != 0) c++;
  if(holes[2][0] != 0) c++;
  return c;
}
```

ir

```
void IR_CONTROL() {
  if (ir.decode(&irr)){
    switch (irr.value) {
      case IRC_RESET:
        //digitalWrite(PIN_RESET,LOW);
        resetFunc();
      break;

      case IRC_MUTE:
        mute=!mute;
        buzzer.stop();
      break;

      case IRC_MANUAL_LIGHT_ENABLE:
        display_manual_light_enable = !display_manual_light_enable;
        DISPLAY_SET_LIGHT(!display_light);
      break;

      case IRC_MANUAL_LIGHT_SET:
        if(display_manual_light_enable)
          DISPLAY_SET_LIGHT(!display_light);
      break;

      case IRC_TOP:
        if(started_game && car_position > 1) car_position--;
      break;

      case IRC_BOT:
        if(started_game && car_position < 3) car_position++;
      break;

      case IRC_START:
        if(!started_game) started_game=!started_game;
      break;

      default:
      break;
    }
    //Serial.println(irr.value);
    ir.resume();
  }
}
```

light

```
int GET_LIGHT_SENSOR_VALUE() {
  return analogRead(PIN_LIGHT_SENSOR);
}

void DISPLAY_SET_LIGHT(bool set) {
  if(set) {
      digitalWrite(PIN_LCD_LIG,LOW);
      display_light = true;
  } else {
      digitalWrite(PIN_LCD_LIG,HIGH);
      display_light = false;
  }
}

void DISPLAY_AUTO_LIGHT() {
  if(display_light) {
    if(GET_LIGHT_SENSOR_VALUE() >= 25)
      DISPLAY_SET_LIGHT(0);
  } else {
    if(GET_LIGHT_SENSOR_VALUE() <= 15)
      DISPLAY_SET_LIGHT(1);
  }
}

void DISPLAY_LIGHT() {
     if(!display_manual_light_enable)
      DISPLAY_AUTO_LIGHT();
}
```

score

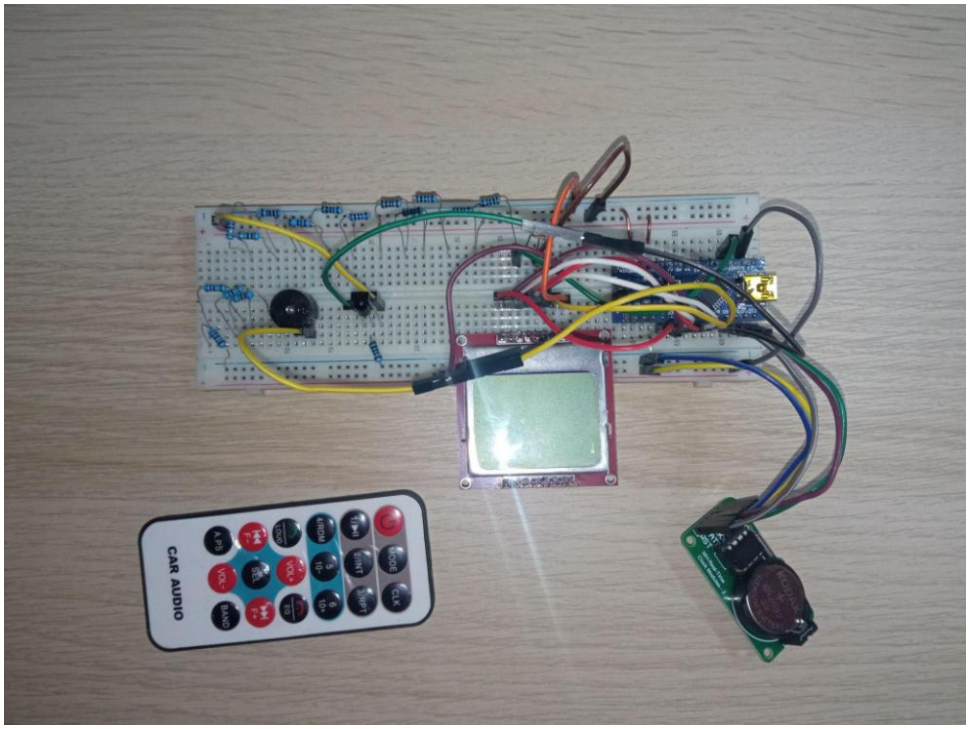
```
void _score_set_timestart() {
  rt.updateTime();
  time_start.Second = rt.seconds;
  time_start.Hour = rt.hours;
  time_start.Minute = rt.minutes;
  time_start.Day = rt.dayofmonth;
  time_start.Month = rt.month;
  time_start.Year = rt.year;
  time_start_unix =  makeTime(time_start);
}

void _score_set_timenow() {
  rt.updateTime();
  time_now.Second = rt.seconds;
  time_now.Hour = rt.hours;
  time_now.Minute = rt.minutes;
  time_now.Day = rt.dayofmonth;
  time_now.Month = rt.month;
  time_now.Year = rt.year;
  time_now_unix =  makeTime(time_now);
}

int _get_score() {
    _score_set_timenow();
    score_tmp = (time_now_unix - time_start_unix);
    score_tmp =   sqrt(score_tmp)*log(score_tmp);
}

void _score() {
    _get_score();
    ltoa(score_tmp,score,10);
    lcd.LCDString("Wynik: ");
    lcd.LCDString(score);
}
```

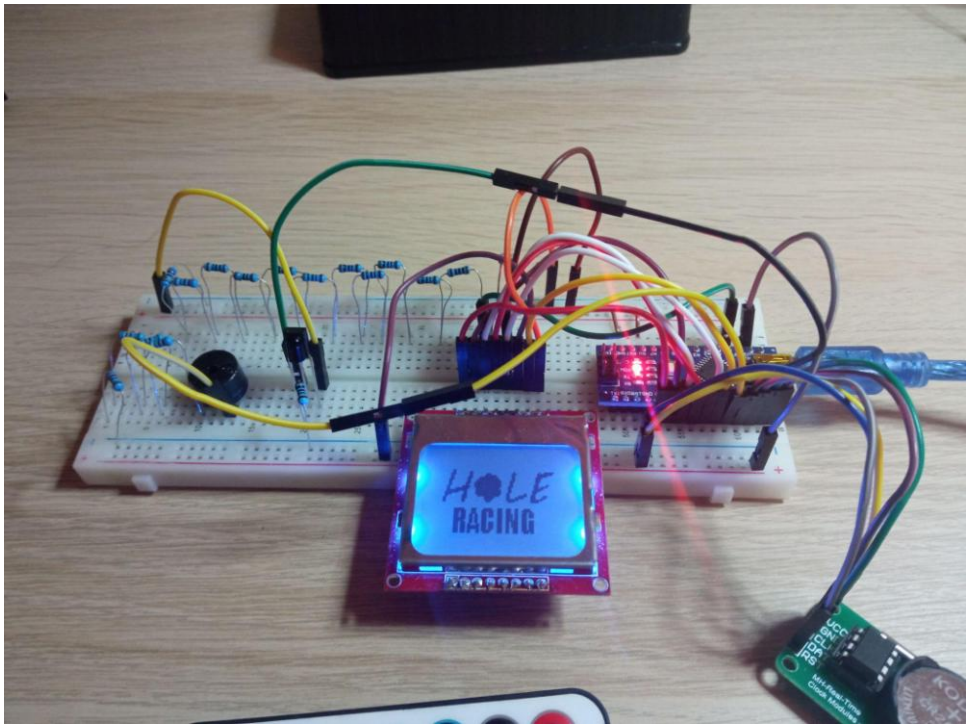
sound

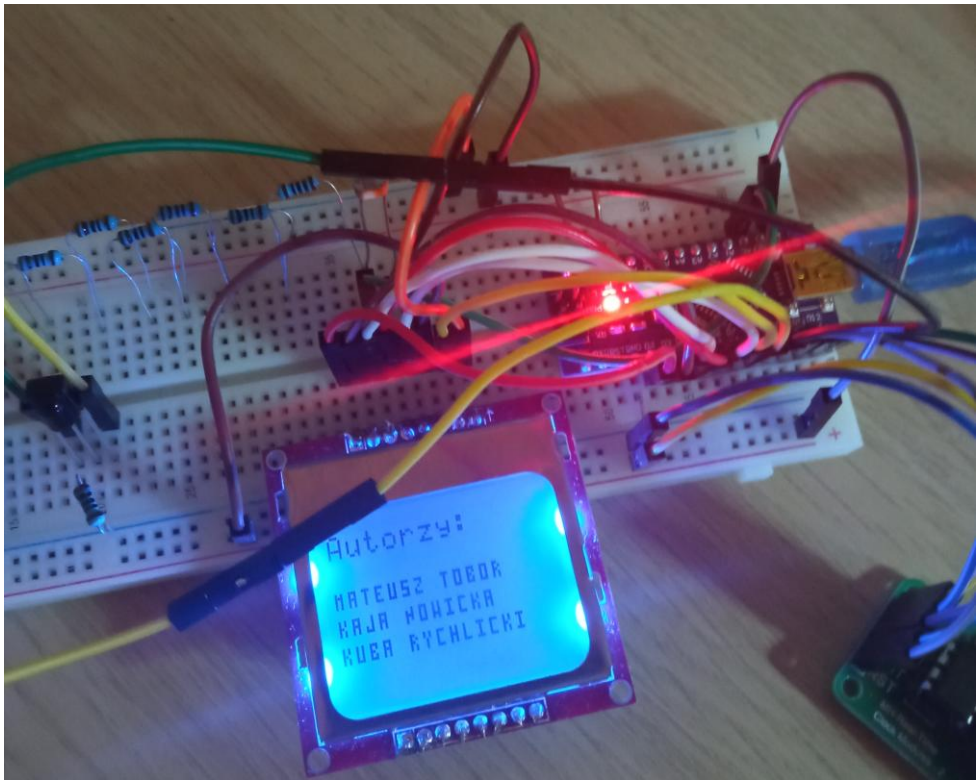
```
int menu_melody[] = {
  NOTE_D5, NOTE_B4, NOTE_D5, NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4,
NOTE_A4, NOTE_FS5, NOTE_E5, NOTE_D5, NOTE_CS5, NOTE_D5, NOTE_CS5,
NOTE_A4, NOTE_D5, NOTE_B4, NOTE_D5, NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4,
NOTE_B4, NOTE_B4, NOTE_G4, NOTE_B4, NOTE_A4, NOTE_B4, NOTE_A4, NOTE_D4,
NOTE_D5, NOTE_B4, NOTE_D5, NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_A4,
NOTE_FS5, NOTE_E5, NOTE_D5, NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4,
NOTE_D5, NOTE_B4, NOTE_D5, NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_B4,
NOTE_B4, NOTE_G4, NOTE_B4, NOTE_A4, NOTE_B4, NOTE_A4, NOTE_D4, NOTE_D4,
NOTE_FS4, NOTE_E4, NOTE_D4, NOTE_E4, NOTE_FS4, NOTE_D4, NOTE_D4,
NOTE_FS4, NOTE_F4, NOTE_D4, NOTE_F4, NOTE_E4, NOTE_D5, NOTE_B4, NOTE_D5,
NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_A4, NOTE_FS5, NOTE_E5,
NOTE_D5, NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_D5, NOTE_B4, NOTE_D5,
NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_B4, NOTE_B4, NOTE_G4, NOTE_B4,
NOTE_A4, NOTE_B4, NOTE_A4, NOTE_D4, NOTE_D5, NOTE_B4, NOTE_D5, NOTE_CS5,
NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_A4, NOTE_FS5, NOTE_E5, NOTE_D5,
NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_D5, NOTE_B4, NOTE_D5,
NOTE_CS5, NOTE_D5, NOTE_CS5, NOTE_A4, NOTE_B4, NOTE_B4, NOTE_G4, NOTE_B4,
NOTE_A4, NOTE_B4, NOTE_A4, NOTE_D4, NOTE_D4, NOTE_FS4, NOTE_E4, NOTE_D4,
NOTE_E4, NOTE_FS4, NOTE_D4, NOTE_D4, NOTE_FS4, NOTE_F4, NOTE_D4, NOTE_F4,
NOTE_E4, NOTE_E4, NOTE_A4, NOTE_CS5, NOTE_FS5, NOTE_E5, NOTE_D5, NOTE_A5
};
int menu_noteDurations[] {

8,4,8,4,8,4,2,8,8,4,8,4,8,4,2,8,4,8,4,8,4,2,8,8,4,8,4,8,4,2,8,4,8,4,8,4,2
,8,8,4,8,4,8,4,2,8,4,8,4,8,4,2,8,8,4,8,4,8,4,8,8,8,1,8,8,1,8,8,8,1,8,8,1,
8,4,8,4,8,4,2,8,8,4,8,4,8,4,2,8,4,8,4,8,4,2,8,8,4,8,4,8,4,2,8,4,8,4,8,4,2
,8,8,4,8,4,8,4,2,8,4,8,4,8,4,2,8,8,4,8,4,8,4,8,8,8,1,8,8,1,8,8,8,1,8,8,8,
2,8,8,8,4,8,4
};

int car_melody[] {
  NOTE_A2,NOTE_B1,NOTE_C2,NOTE_D3
};

int car_noteDurations[] {
  16,8,8,16
};

int lost_melody[] {
  NOTE_D7,NOTE_D7,NOTE_A6,NOTE_D7,NOTE_B6
  //NOTE_A2,NOTE_B1,NOTE_C2,NOTE_D3
};

int lost_noteDurations[] {
  4,4,4,4,4
};

//-------------------------------------------------------------------------
-------------------------------------------------------------------------
```

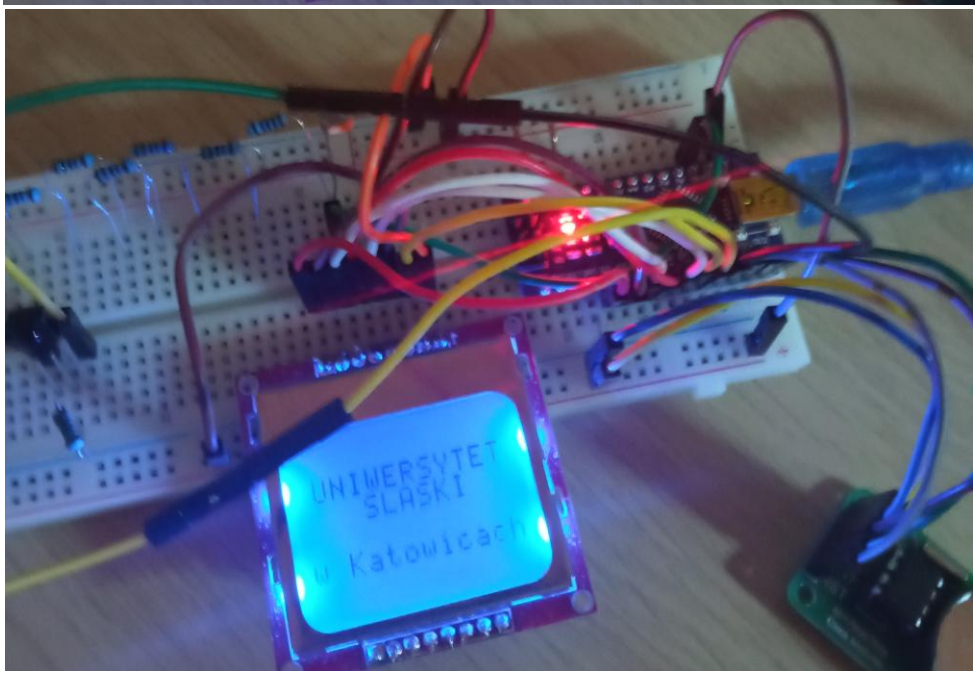
```
------------------

void sound_menu() {
  int noteLength = sizeof(menu_noteDurations) / sizeof(int);
  if(buzzer.getState() == BUZZER_IDLE)
    buzzer.playMelody(menu_melody, menu_noteDurations, noteLength); //
playing
}

int sound_car() {
  int noteLength = sizeof(car_noteDurations) / sizeof(int);
  if(buzzer.getState() == BUZZER_IDLE)
    buzzer.playMelody(car_melody, car_noteDurations, noteLength); //
playing
}

int sound_lost() {
  int noteLength = sizeof(lost_noteDurations) / sizeof(int);
  if(buzzer.getState() == BUZZER_IDLE)
    buzzer.playMelody(lost_melody, lost_noteDurations, noteLength); //
playing
}
```

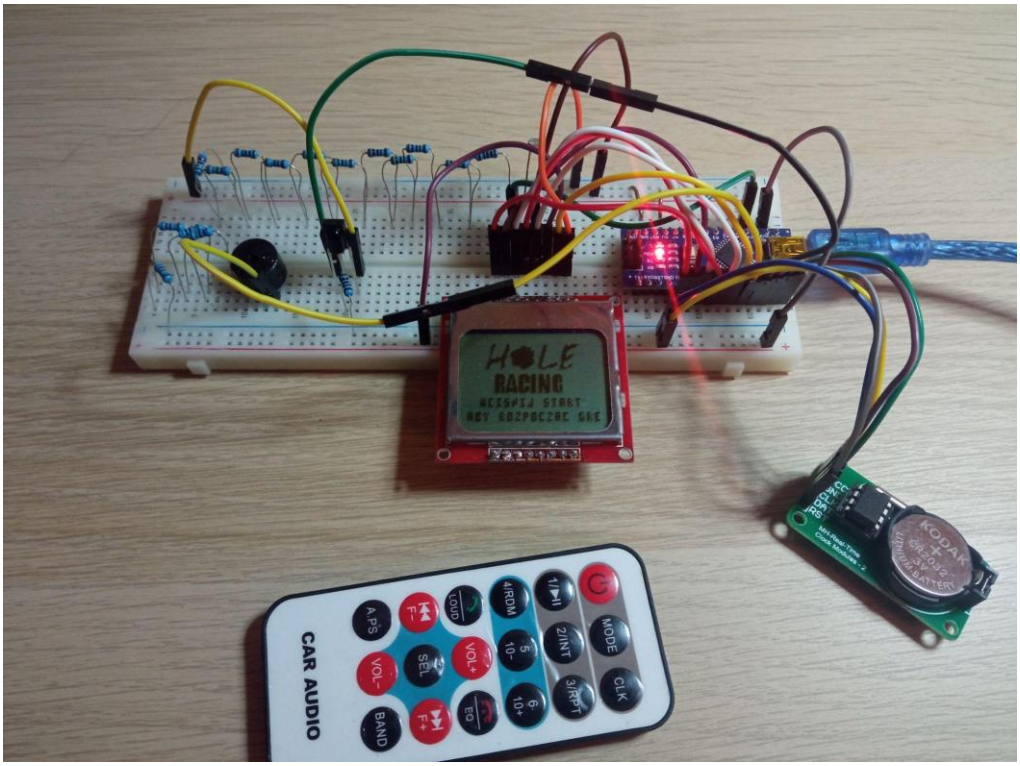
Zdjęcia przykładowego wykonania

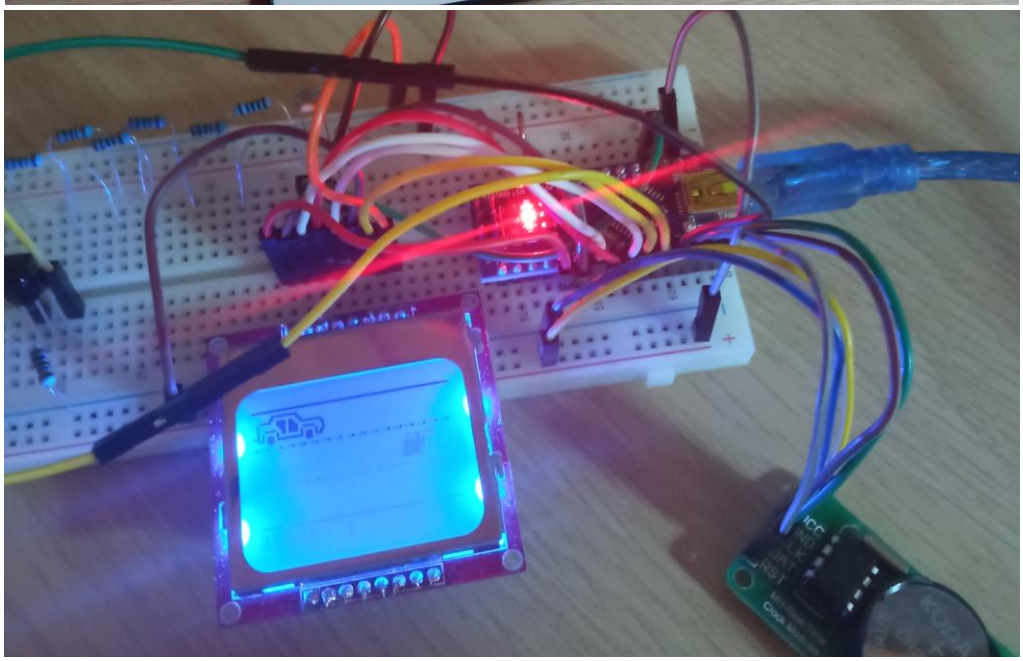


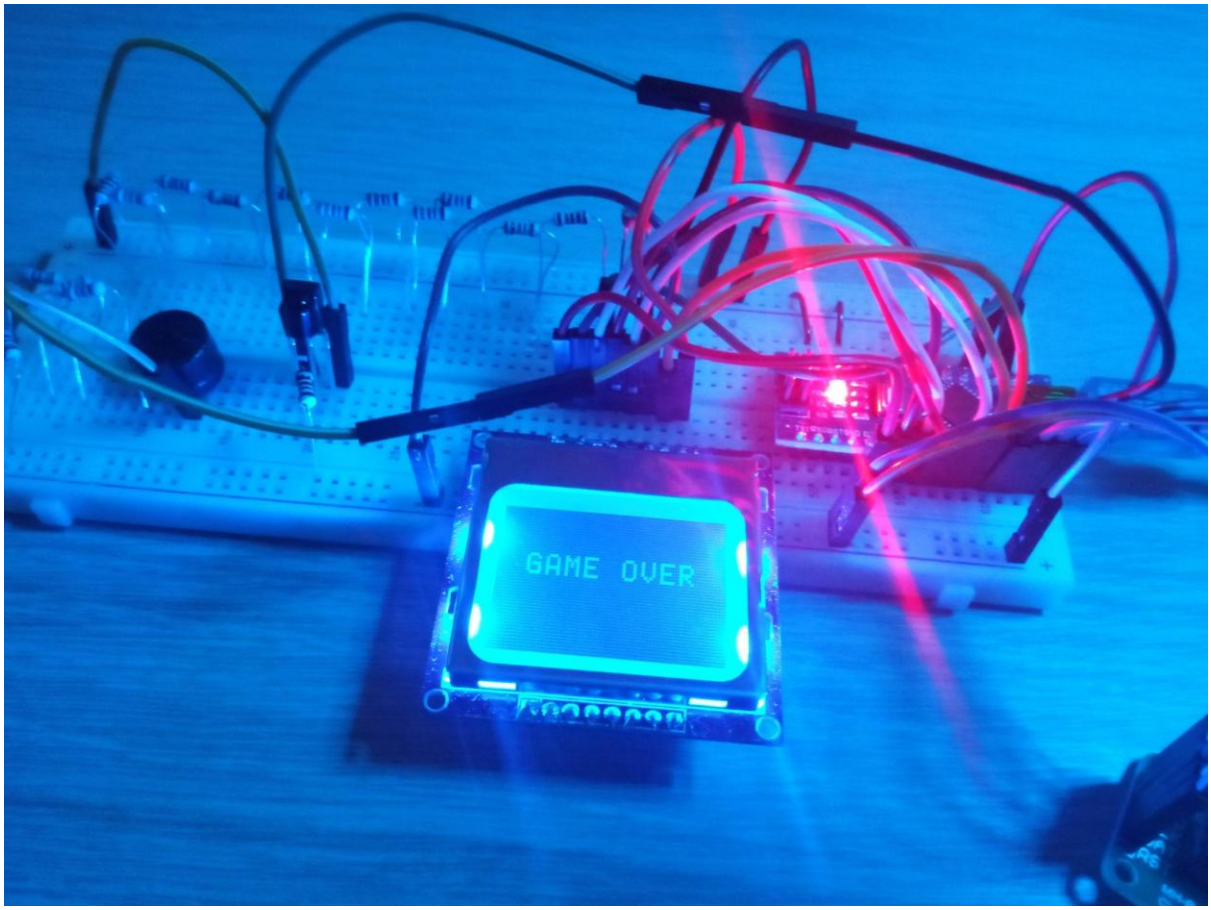



Autorzy:
MATEUSZ TOBOR
KAJA NOWICKA
KUBA RYCHLICKI


UNIWERSYTET
SLASKI


HOLE
RACING
CAR AUDIO
A.PS
LOUD
VOL+
VOL-
SEL
BAND
EQ
MODE
CLK
KODAK





Film z prezentacją projektu

Film z prezentacją projektu dostępny pod adresem:
https://www.youtube.com/watch?v=ZC0TMcRZ5ps